

CURAC 2014

Synergie durch Interdisziplinarität

11. - 13. September 2014
München

Tagungsband

H. Feußner (Herausgeber)



13. Jahrestagung der Deutschen
Gesellschaft für Computer- und Roboterassistierte Chirurgie



www.curac.com

ISBN 978-3-00-047154-4

13. Jahrestagung der Deutschen Gesellschaft für Computer- und Roboterassistierte Chirurgie e. V.

Tagungspräsident: Prof. Dr. Hubertus Feußner

zusammen mit

Forschungsgruppe für Minimal-invasive Interdisziplinäre Therapeutische Intervention (MITI)

Allgemeine Leitung: Prof. Dr. Hubertus Feußner

Wissenschaftlicher Leiter: Dr.-Ing. Armin Schneider

Klinischer Leiter: PD Dr. Dirk Wilhelm



Sektion für minimal-invasive Computer- und Telematik-assistierte Chirurgie der DGCH (CTAC)

1. Vorsitzender: Prof. Dr. Hubertus Feußner

2. Vorsitzender: PD Dr. Markus Kleemann



Sonderforschungsbereich 125 „Cognition-Guided Surgery“

Sprecher: Prof. Dr. Markus Büchler

Wissenschaftlicher Sekretär: Prof. Dr. Beat Müller

Stellv. wissenschaftl. Sekretär: Dr. Hannes Kenngott





**FORSCHUNGSGRUPPE FÜR
MINIMAL-INVASIVE INTERDISZIPLINÄRE
THERAPEUTISCHE INTERVENTION**



Klinikum rechts der Isar
Technische Universität München
Chirurgische Klinik und Poliklinik

Herausgeber:

Prof. Dr. Hubertus Feußner
Chirurgische Klinik und Poliklinik
Klinikum rechts der Isar
Technische Universität München
Ismaninger Straße 22
81675 München

Tel: 089/4140-2030
Fax: 089/4140-6030
Email: hubertus.feussner@tum.de

H. Feußner (Herausgeber)

CURAC 2014

Synergie durch Interdisziplinarität

13. Jahrestagung der Deutschen Gesellschaft
für Computer- und Roboterassistierte Chirurgie

11. - 13. September 2014
München



Tagungsband

Design and Implementation of a Task Manager Prototype for the Operation Room

M. Wiemuth¹, O. Burgert¹

¹ Reutlingen University, Medical-Technical Informatics, Reutlingen, Germany

Contact: markus.wiemuth@reutlingen-university.de

Abstract:

An operation room is a stressful work environment. Nevertheless, all involved persons have to work safely as there is no space for making mistakes. To ensure a high level of concentration and seamless interaction, all involved persons have to know their own tasks and tasks of their colleagues. The entire team must work synchronously at all times. However, the operation room (OR) is a noisy environment and the actors have to set their focus on their work. To optimize the overall workflow, a task manager supporting the team was developed. Each actor is equipped with a client terminal showing a summary of their own tasks. Moreover, a big screen displays all tasks of all actors. The architecture is a distributed system based on a communication framework that supports the interaction of all clients with the task manager. A prototype of the task manager and several clients have been developed and implemented. The system represents a proof-of-concept for further development. This paper describes the concept of the task manager.

Keywords: task manager, workflow, surgical workflow, integrated operation room

1 Problem

Working in an OR is noisy, expensive and stressful. One minute of using an OR costs about 12 euro [1]. But at the same time, all team members in the OR have to be very concentrated and have to synchronize their work pre- intra- and post-operatively. Therefore, communication of individual progress to other team members plays a crucial role to avoid waiting times or potentially dangerous situations. Furthermore, in other disciplines, like aviation, checklist driven approaches have helped reducing risks and stress [2] [3].

Currently, the team members in the OR have no system which supports guidance on task level. They have to speak up to communicate their status, or they rely on their experience. Both methods are error prone and may increase the noise level and stress factor. At the same time it might happen, that some team member does not notice the state of other team member or some forget to announce their state. The team is not synchronized and the work in the OR can be delayed, or risk situations might occur. Currently most work in the area of workflow management in the OR was done only considering the surgeons' work [4] [5]. Focusing on the whole team in the OR is important to keep the team synchronized.

The aim of the work is to design and implement a task manager, supporting the whole team in the OR and displaying the state of every involved actor. We assume the process to be already sub-divided into tasks, which can be executed in more or less sequential order by each actor. The concept of workflow managers in the OR was used to design the task manager [6] [7] [8].

2 Methods

The main goal is to give the actors in and outside the OR an overview of the state of every involved actor. We assume, that an integrated OR system is already available, and the prototype has to interact with the OR system, e.g. by receiving information or by using the input and output devices already available. For the first prototype of the task manager, the key requirements shown in Table 1 were defined together with our industrial partner KARL STORZ:

Requirement	Description
Multi-actor support	A multi-actor support is necessary, because every actor shall have the opportunity to interact with the system.
Distributed System	A distributed system is needed to connect the interfaces for the actors.
Interaction	Every actor shall have their own interface to interact with the task manager.
Cross platform	The conventional platforms should be supported to get a large number of possible interfaces.
Communication	The protocol works with a KARL STORZ communication framework.
Interface	Every actor can have only one interface. More than one terminal for a role in the OR is not a requirement.
Overall view	There shall be a centralized overview of the state of all team members.
Data logging	All interactions with the system shall be saved and documented.

Table 1: Key requirements of the prototype.

Based on the key requirements, we decided to build a system consisting of a big screen which shows the tasks of all actors, and personalized client terminals (handhelds or stationary), which are used for user input and display of each actor's tasks. These client terminals and the big screen are connected to a task manager server, which takes over the distribution of all messages in the network and keeps the clients synchronous.

Each "process" consists of "tasks", each task has a "state". In the beginning, every task is in state "open". All tasks of the actors, which are in progress, are in the state "process". A task is in state "finished", if it is completed by the personnel. The state "skipped" indicates a task was skipped by one of the actors. The abstraction level of the task depends on the user needs, the possibility to detect the tasks and the factor to not overstrain the actors.

The visualization on the central task board screen, which shows an overview of all tasks, is based on the representation of the scrum board [9]. The advantage of this is to display all necessary data comprehensible and compact. Figure 1 shows the concept design of the task board. All actors, who participate in the operation, are represented in one column with their task. The last three columns used to display all tasks, which are currently in process, finished or skipped.

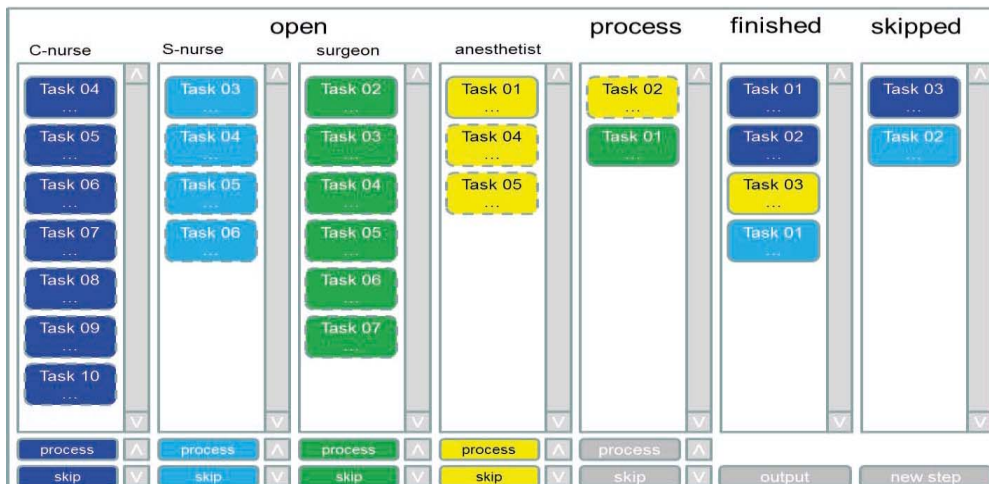


Figure 1: Concept of the task board

Every actor who needs to interact with the system, gets their own client terminals. Those devices run an instance of the task manager client. The clients or interfaces can contain different information, depending on the needs of the user. In order to be able to evaluate the usability of different client interaction approaches with different devices, three different graphical interfaces were designed. Their description can be found in Table 2.

Interface	Description
Small client	Shows the active and the next step of the user
Normal client	Shows a list of all steps for the user as well as the steps of all users in the states: process, finished and skipped
Task board client	Shows all steps of all users in the states: open, process and skipped

Table 2: The three different interfaces for the actors.

The system is designed as a distributed system with a master node (server) and different clients. The task manager server is running on a computer with the operating system Microsoft Windows 7. The client terminals can run on different devices and operation systems such as smartphones and tablets with Android or Windows or other devices running Linux. The messages are delivered via a communication framework made by KARL STORZ [10]. The task manager sever, as well as the client terminals, are connected to the communication framework via LAN/Wi-Fi and using a TCP/IP connection to deliver information. The connection of the clients is based on the handshaking method. Every client terminal sends a message, contains the type of the client, to the task manager server. The task manager server can accept the connection and sends back a message containing a ClientID. Finally, the client terminal accepts the connection and sends its DeviceID to the task manager server. The task manager server can now identify the client as a unique device and the client can communicate with the server.

3 Results

The Task Manager was developed according to the description above. Since the main task of the master node is to receive and transmit all messages between the client terminals, it was developed without a graphical user interface. All messages are sent to the task manager server, and the task manager server broadcasts them to all client terminals. The task board client shows an overview of all tasks of all actors in the OR. It is implemented as a windows client. At the moment, it is not possible to do user inputs with this client, since its main purpose is to show the needed information to the user.

The client terminals can be different, depending to the actor's needs. For example, the surgeon can use a footswitch to make the interaction with the foot to have free hands during the surgery. The scrub nurse can get a stationary client in her working area. The circulation nurse can get a tablet or smartphone that can be taken to any location. All client terminals can send the user inputs directly to the task manager server and are able to display the changes of all tasks except the footswitch. The footswitch can only be used as an input device without visualization. The state change of the footswitch is delivered to the task manager server, and all other clients can react to the changes. The client hardware (tablet, smartphone or PC), the information shown to the actor and the user interaction depends on the user's needs and preferences. Figure 2 to Figure 5 show an example of every developed terminal.

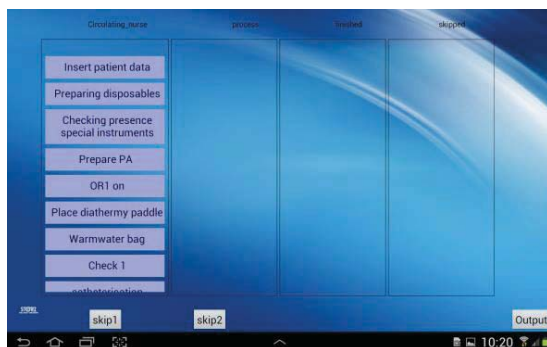


Figure 1: 10'' Android Client



Figure 2: 3.14'' Android Client

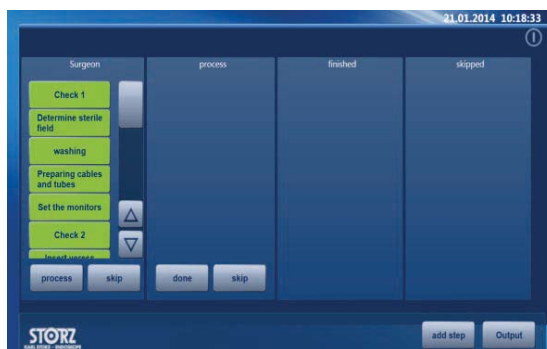


Figure 5: Windows 10'' Client

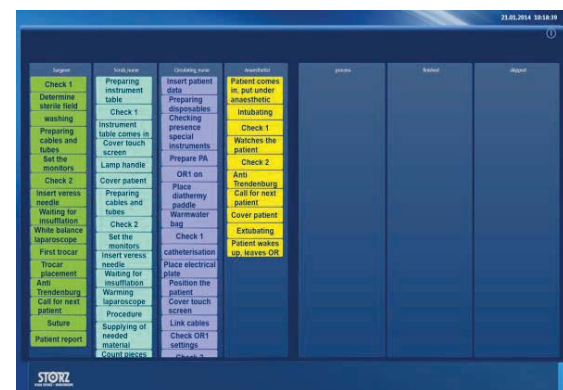


Figure 3: Big screen for the overview

During an operation, every actor gets a client terminal to make their user inputs and to get an overview of all tasks. One big screen is placed in the OR, so that every actor can see it.

Up until now, the system has been successfully tested in a laboratory environment based on the tasks occurring in a real surgical intervention. The chosen intervention is a laparoscopic cholecystectomy. The tasks for the intervention were taken from [11]. Every intervention and the required task can be loaded to the system, but currently we used only the mentioned intervention and tasks.

4 Discussion

In this paper, the concept of the task manager system has been introduced and implemented. With this system, every actor can get the possibility to see what they have to do next as well as what others currently do or will do next. The team gets a new option to keep in sync during an operation. Additionally, every actor can see what others did or have to do during the operation.

The challenge in bringing task support in the OR is to find a balance between user interaction, complexity and level of support. Since we want to limit the needed user interaction to a minimum, we decided to use tasks on a higher abstraction level. We are preparing a clinical evaluation of the system which has to answer the question whether a linear task sequence with the possibility to skip a step is sufficient or if this causes irritations.

After the evaluation, the task manager shall be improved by functions like the automatic state detection for each task sequence. This can be realized using the state of the connected devices and other information in the network.

5 Acknowledgment

This work was supported by the company KARL STORZ GmbH & Co. KG D-Tuttlingen.

6 References

- [1] Bauer, Hartwig (2010): Cockpit und OP-Saal: Checklisten verbessern Sicherheit. Deutsche Gesellschaft für Chirurgie. Online verfügbar unter http://www.dgch.de/fileadmin/media/texte_pdf/servicemeldungen/Sicherheitschecklist_Artikel_Bauer.pdf, zuletzt geprüft am 08.10.2013.
- [2] Kohn, Linda T.; Corrigan, Janet; Donaldson, Molla S. (2000): To err is human. Building a safer health system. Washington, D.C: National Academy Press (Quality chasm series).
- [3] Rall, M.; Lackner, C. K. (2010): Crisis Resource Management (CRM). In: Notfall Rettungsmed 13 (5), S. 349–356. DOI: 10.1007/s10049-009-1271-5.
- [4] Forestier, Germain; Lalys, Florent; Riffaud, Laurent; Louis Collins, D.; Meixensberger, Jurgén; Wassef, Shafik N. et al. (2013): Multi-site study of surgical practice in neurosurgery based on surgical process models. In: Journal of Biomedical Informatics. DOI: 10.1016/j.jbi.2013.06.006.
- [5] Forestier, Germain; Lalys, Florent; Riffaud, Laurent; Trelhu, Brivael; Jannin, Pierre (2012): Classification of surgical processes using dynamic time warping. In: Journal of Biomedical Informatics 45 (2), S. 255–264. DOI: 10.1016/j.jbi.2011.11.002.
- [6] Thomas Neumuth (2012): Surgical Process Modeling. - Theory, Methods, and Applications -. Habilitation thesis, Leipzig. Online verfügbar unter <http://www.iccas.de/wp-content/uploads/2012/12/Neumuth-Surgical-Process-Modeling.pdf>, zuletzt geprüft am 09.10.2013.
- [7] Forestier, Germain; Lalys, Florent; Riffaud, Laurent; Louis Collins, D.; Meixensberger, Jurgén; Wassef, Shafik N. et al. (2013): Multi-site study of surgical practice in neurosurgery based on surgical process models. In: Journal of Biomedical Informatics. DOI: 10.1016/j.jbi.2013.06.006.
- [8] Jannin, P.; Morandi, X. (2007): Surgical models for computer-assisted neurosurgery. In: NeuroImage 37 (3), S. 783–791. DOI: 10.1016/j.neuroimage.2007.05.034.
- [9] Scrum Alliance, Inc.: Scrum Alliance: Internetpräsenz. <http://www.scrumalliance.org/>. Version:25.11.2013
- [10] KARL STORZ GmbH & Co. KG: KARL STORZ GmbH & Co. KG: Homepage. 2013
- [11] Lier, Lotte van (2008): Design of a digital checklist interface for preparing laparoscopic procedures. TUDelft, Delft. Industrial Design.